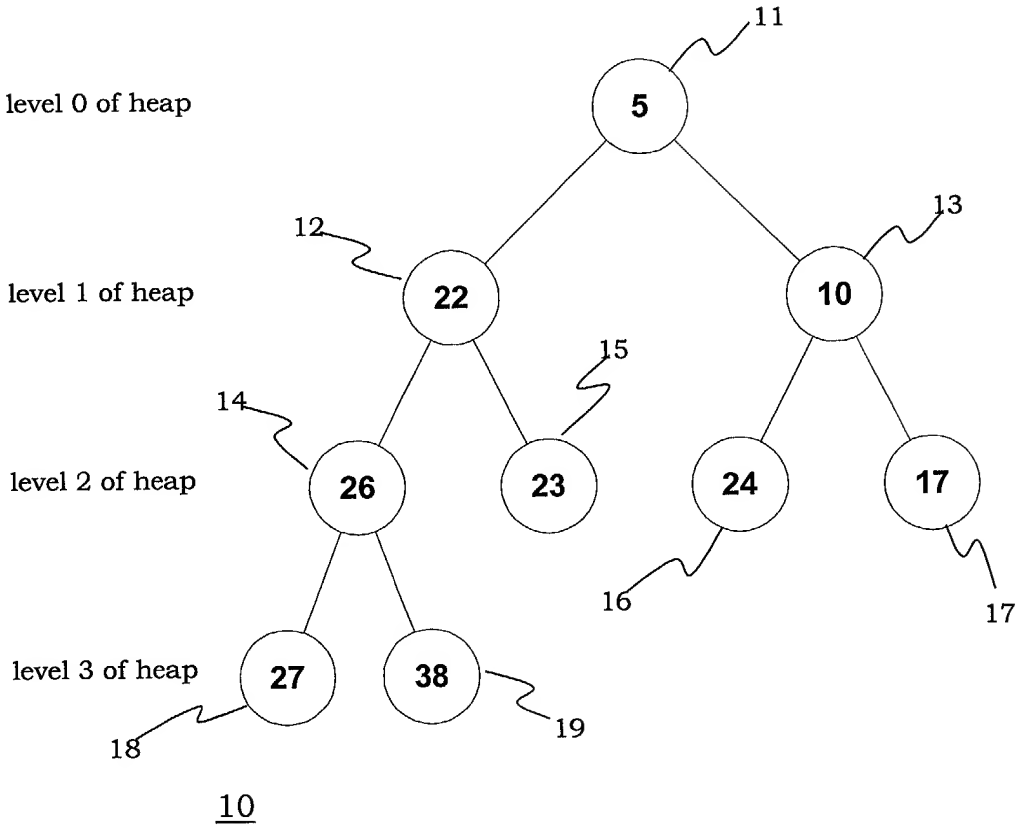


FIGURE 1
(Prior Art)

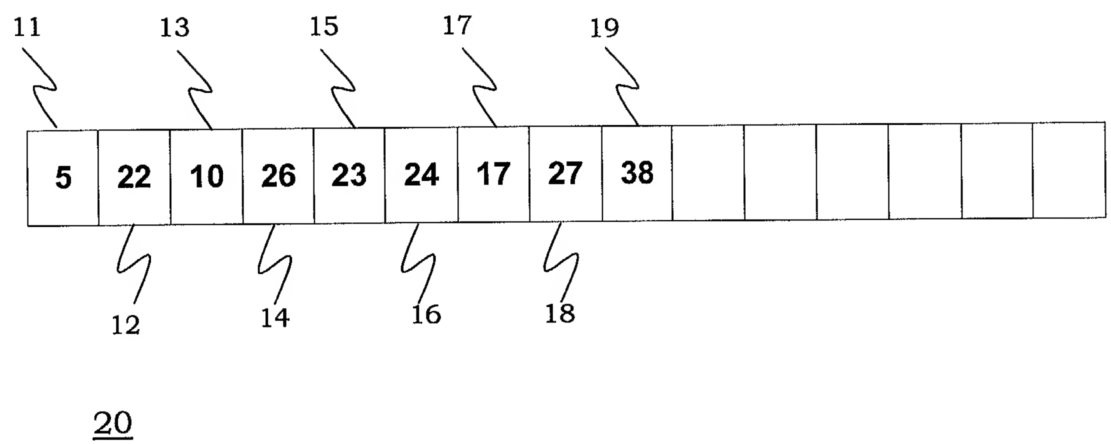


FIGURE 2
(Prior Art)

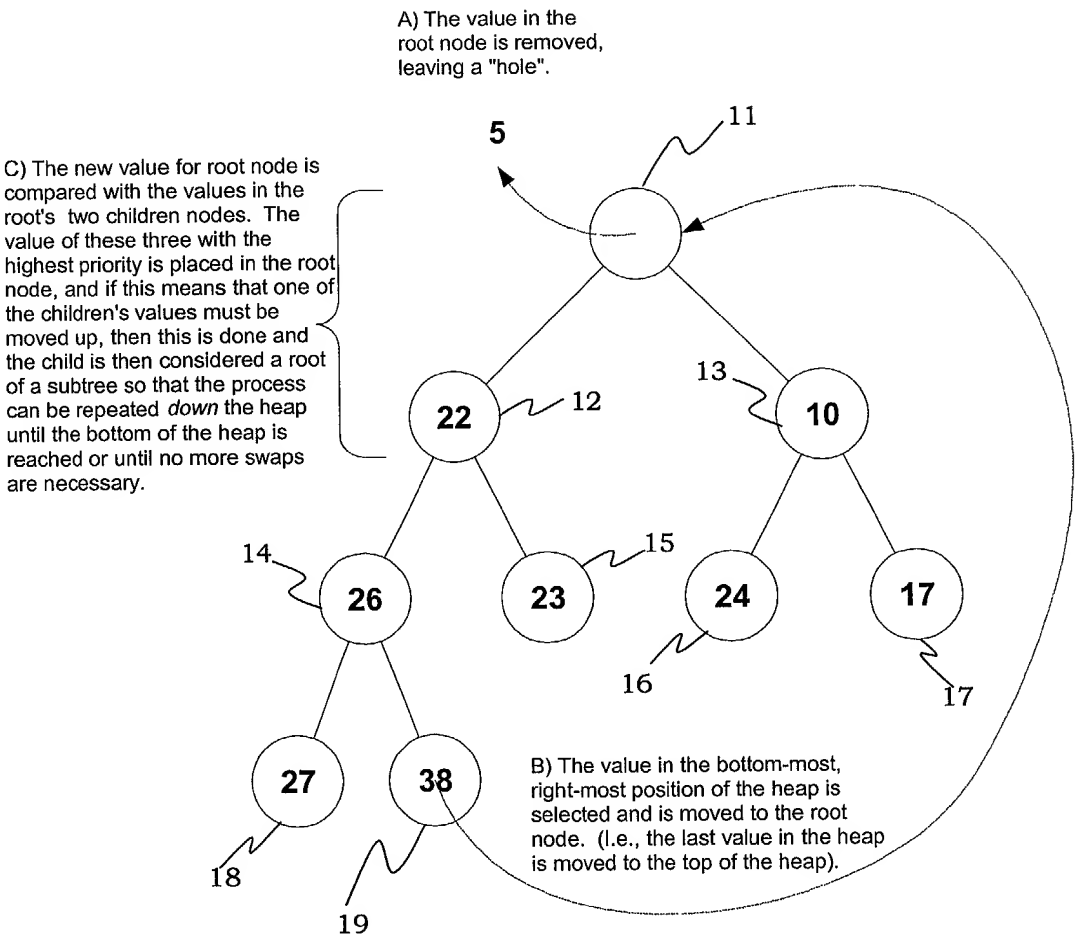
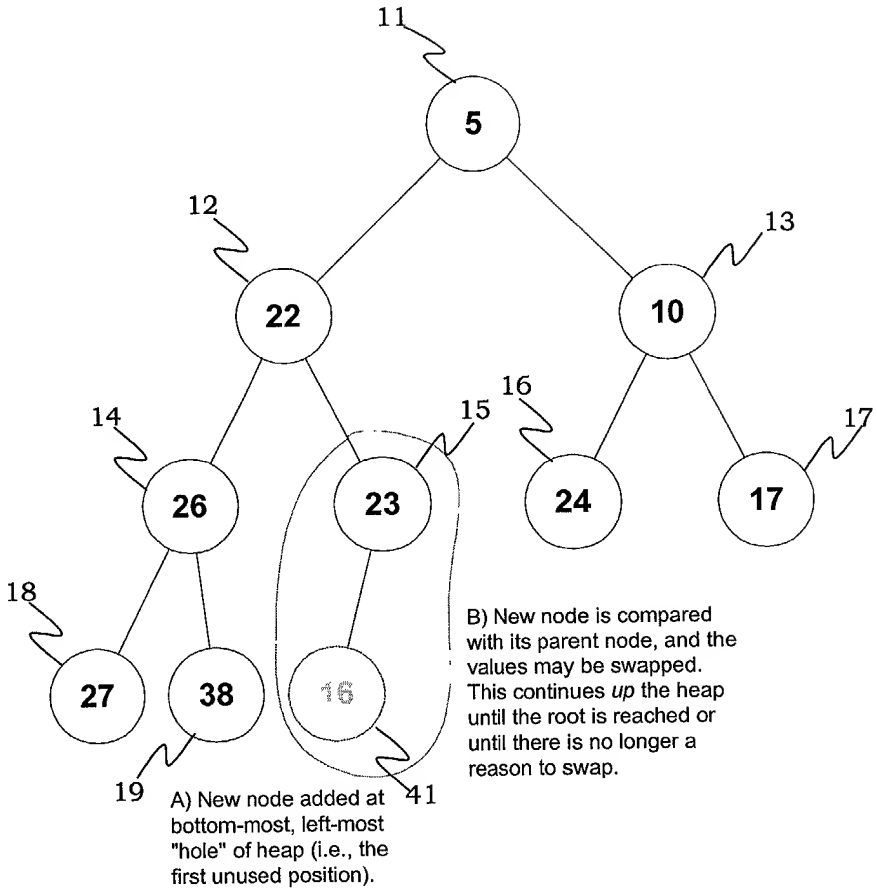
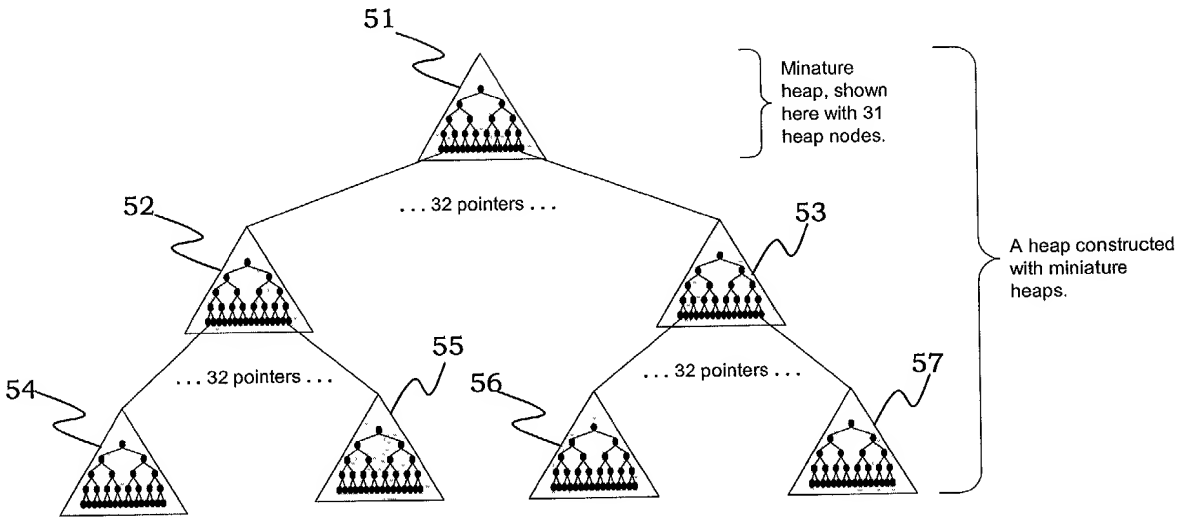


FIGURE 3
(Prior Art)



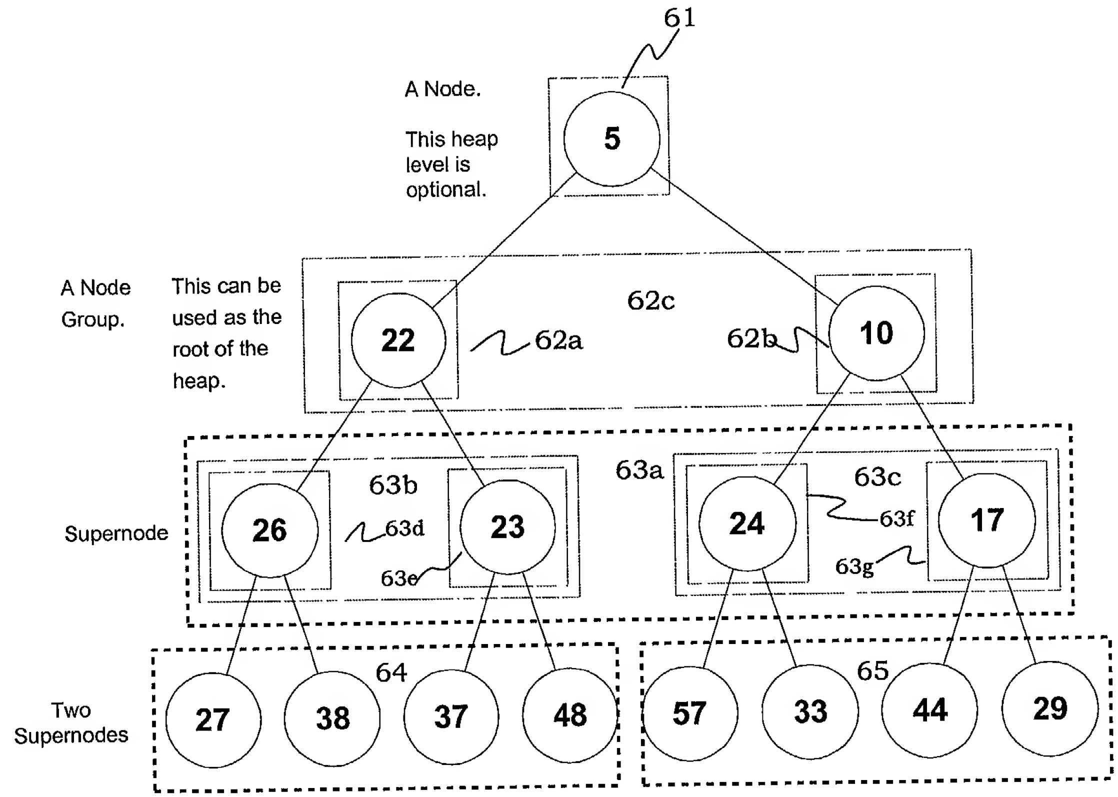
40

FIGURE 4
(Prior Art)


51
Minature heap, shown here with 31 heap nodes.
52
. . . 32 pointers . . .
53
A heap constructed with miniature heaps.
55
. . . 32 pointers . . .
54
. . . 32 pointers . . .
56
57

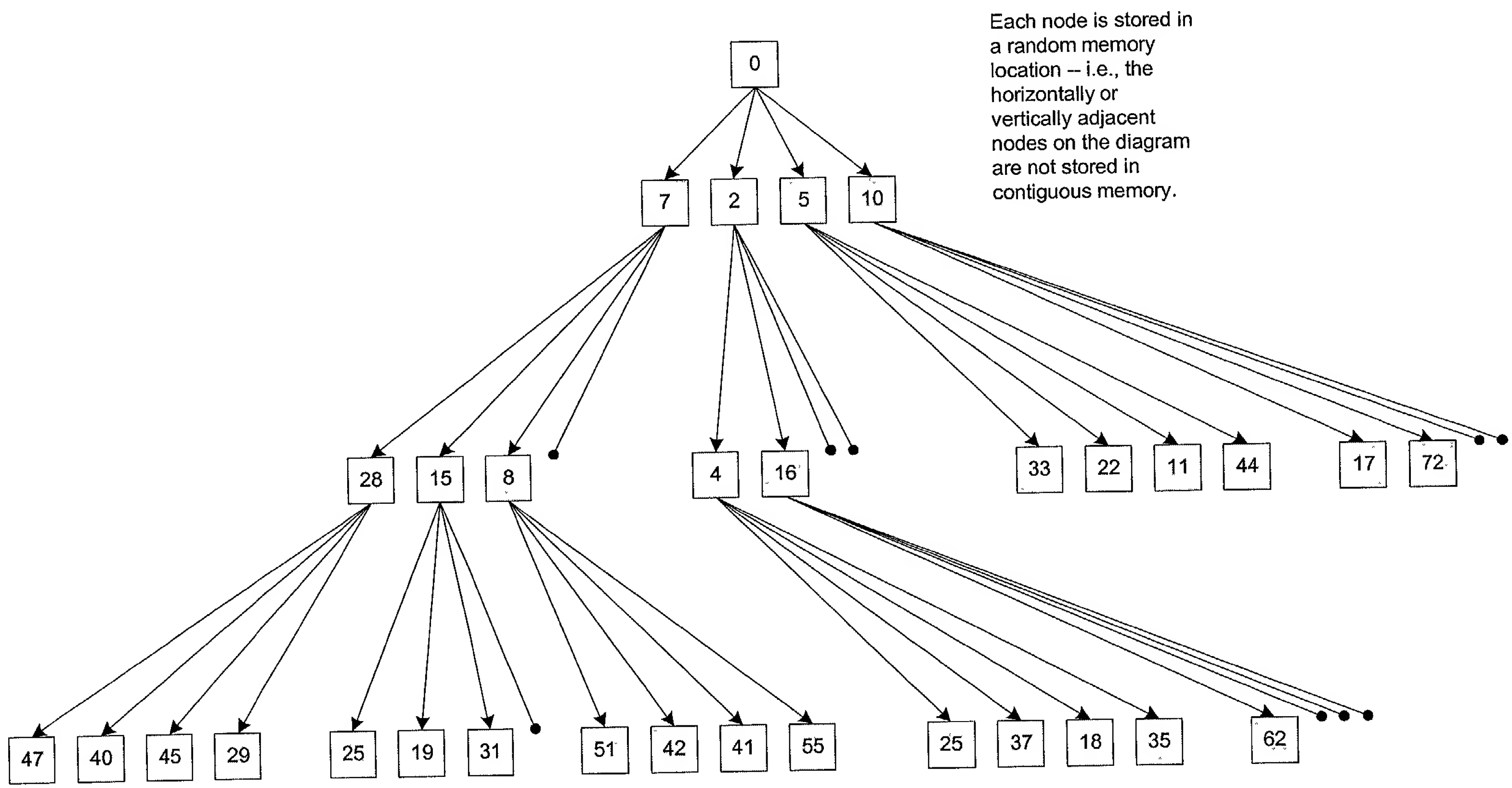
50

FIGURE 5


61
A Node.
This heap level is optional.
5
A Node Group.
This can be used as the root of the heap.
22
62a
62c
62b
10
Supernode
26
63b
63d
63e
23
63a
24
63c
63f
63g
17
Two Supernodes
64
27
38
37
48
65
57
33
44
29

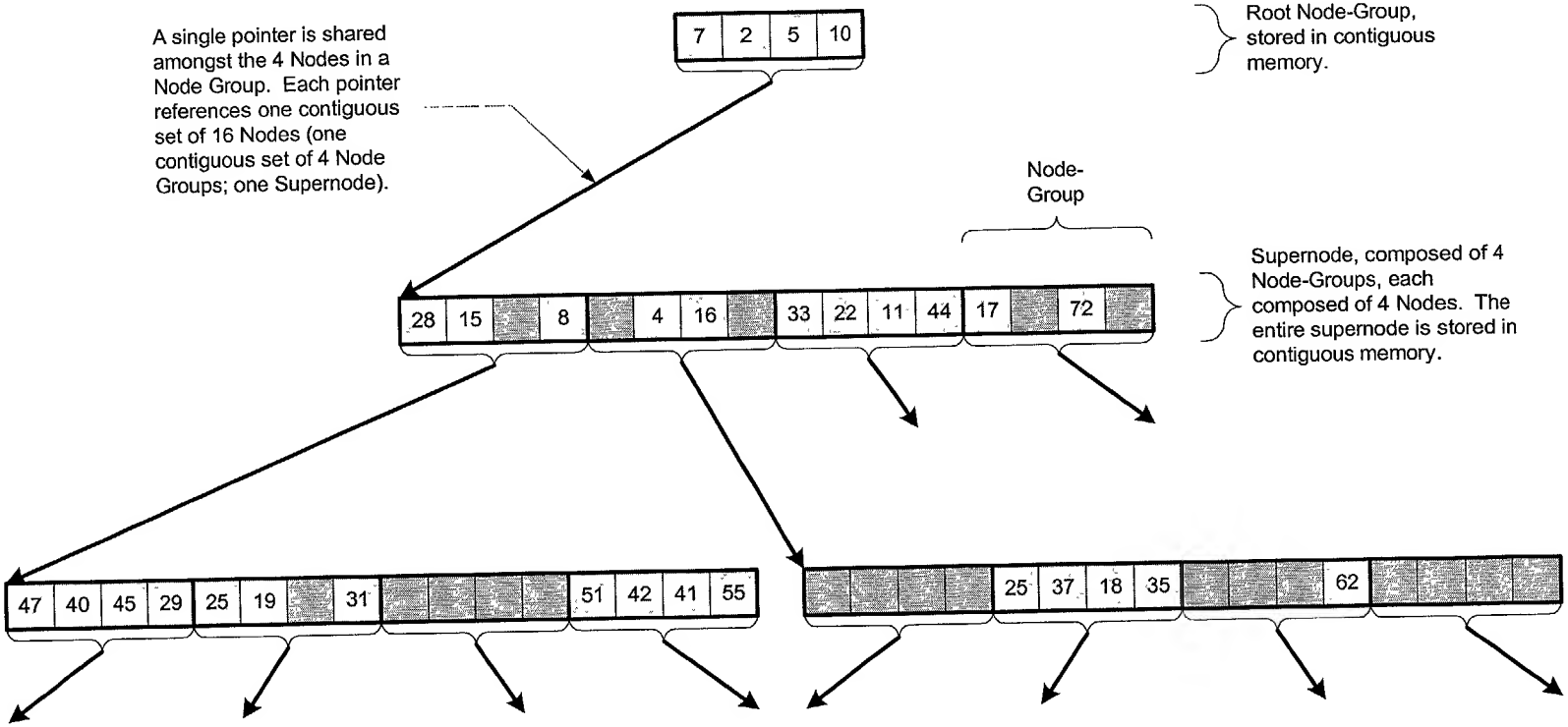
60

FIGURE 6


Each node is stored in a random memory location -- i.e., the horizontally or vertically adjacent nodes on the diagram are not stored in contiguous memory.
0
7
2
5
10
28
15
8
4
16
33
22
11
44
17
72
47
40
45
29
25
19
31
51
42
41
55
25
37
18
35
62

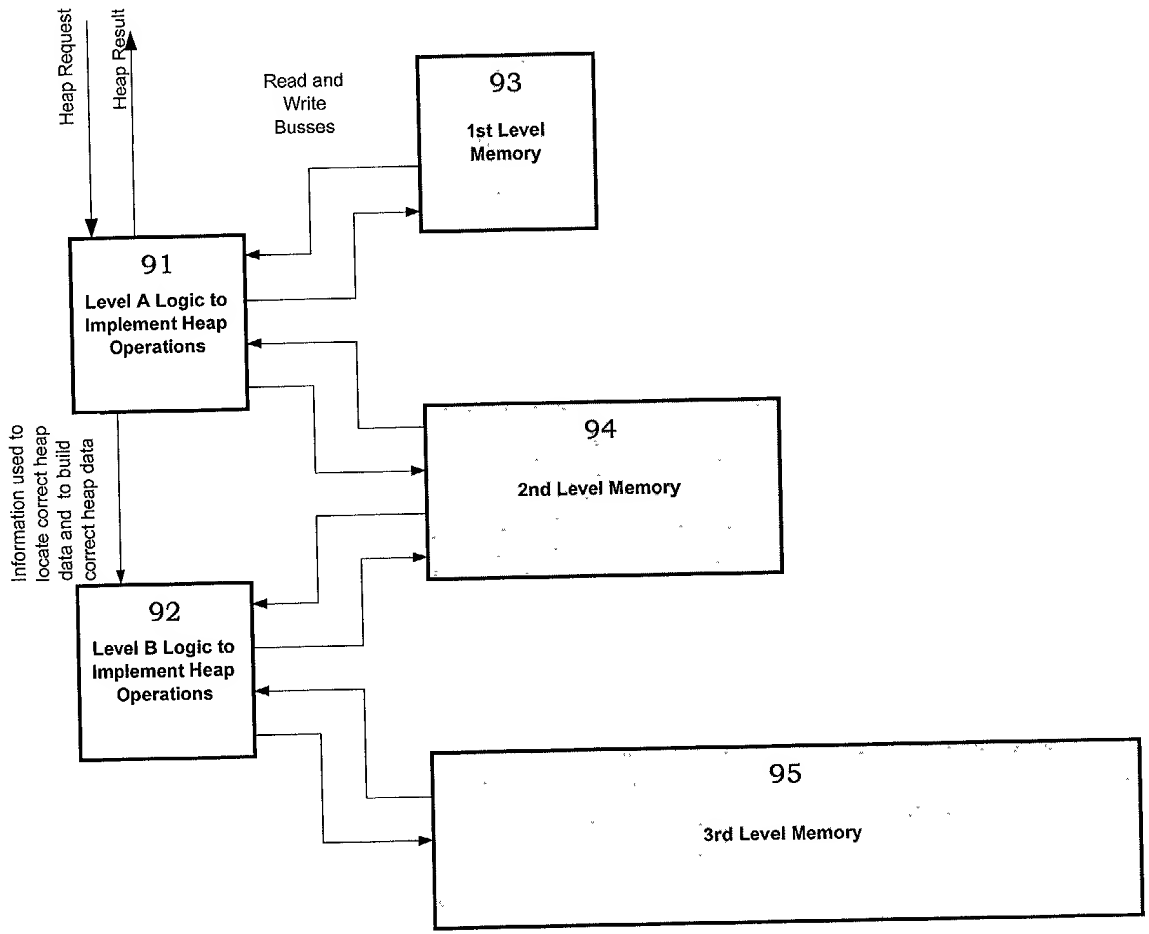
70

FIGURE 7


A single pointer is shared amongst the 4 Nodes in a Node Group. Each pointer references one contiguous set of 16 Nodes (one contiguous set of 4 Node Groups; one Supernode).
7 2 5 10
Root Node-Group, stored in contiguous memory.
Node-Group
28 15 8 4 16 33 22 11 44 17 72
Supernode, composed of 4 Node-Groups, each composed of 4 Nodes. The entire supernode is stored in contiguous memory.
47 40 45 29 25 19 31 51 42 41 55
25 37 18 35 62

80

FIGURE 8

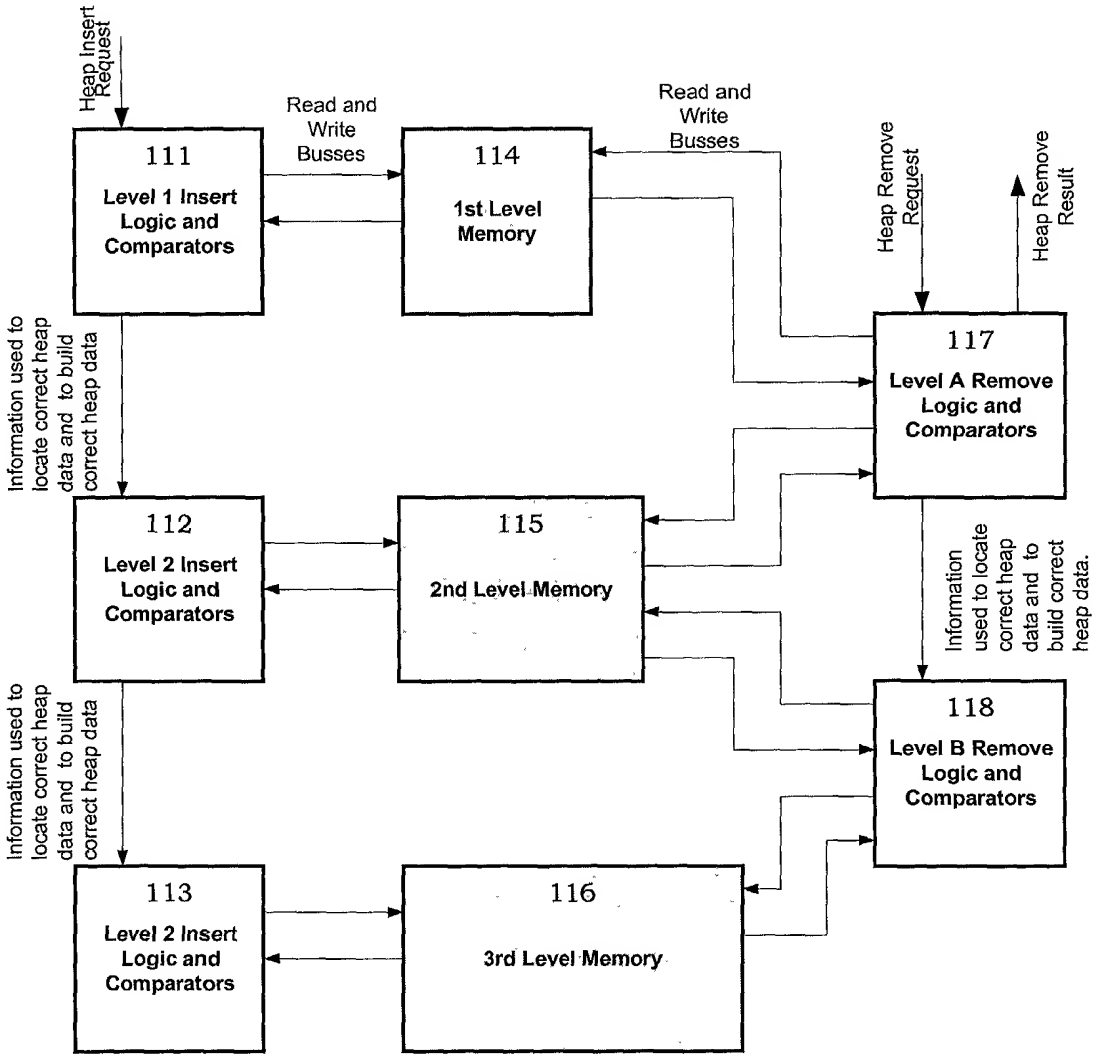

Heap Request
Heap Result
Read and Write Busses
93
1st Level Memory
91
Level A Logic to Implement Heap Operations
94
2nd Level Memory
Information used to locate correct heap data and to build correct heap data
92
Level B Logic to Implement Heap Operations
95
3rd Level Memory

90

FIGURE 9

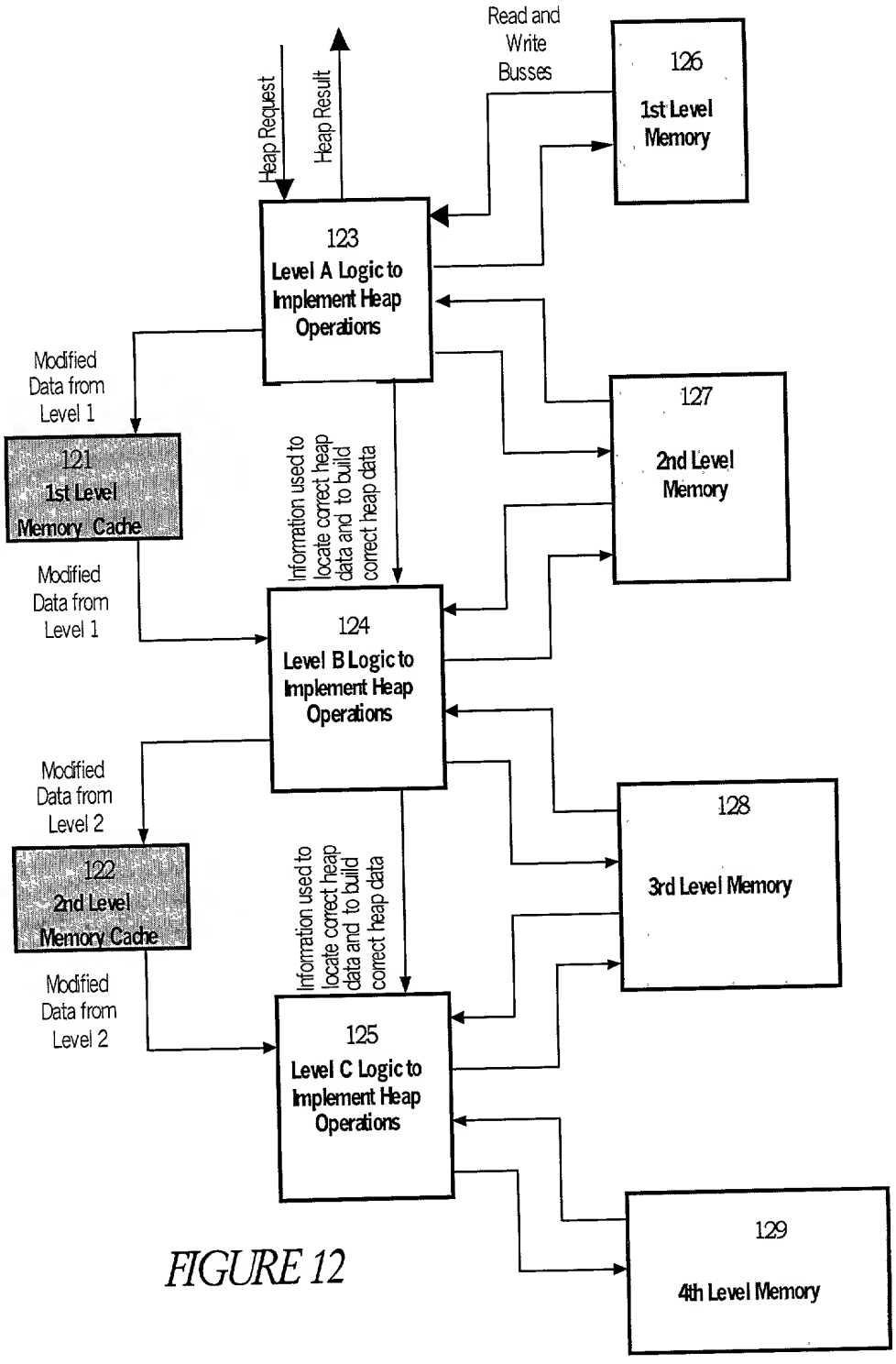
	time ---------->																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Read Level 1 RAM	A						B											
Write Level 1 RAM						A						B						
Level A Comparisons				A	A					B	B							
Read Level 2 RAM			A						B									
Write Level 2 RAM								A						B				
Level B Comparisons						A	A					B	B					
Read Level 3 RAM					A						B							
Write Level 3 RAM										A						B		
Level C Comparisons								A	A					B	B			
Read Level 4 RAM							A						B					
Write Level 4 RAM										A						B		

FIG. 10



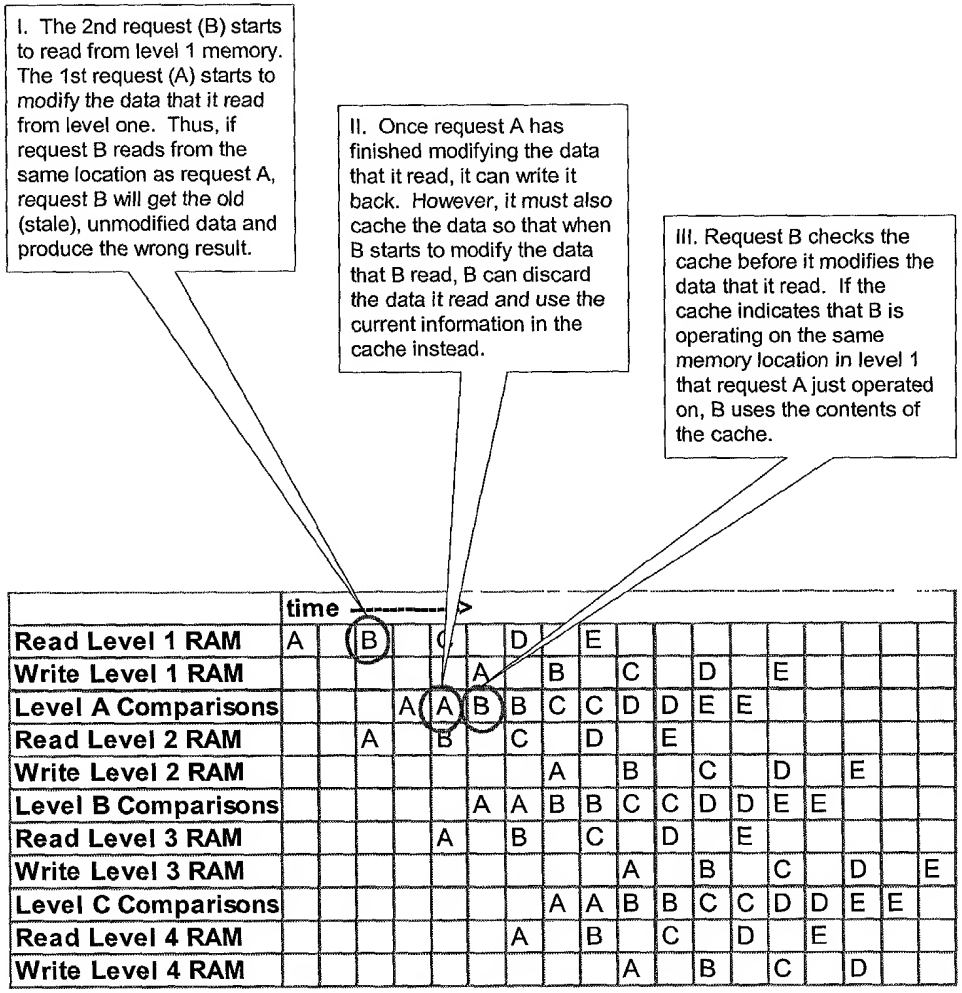
110

FIGURE 11



120

FIGURE 12



130

FIGURE 13

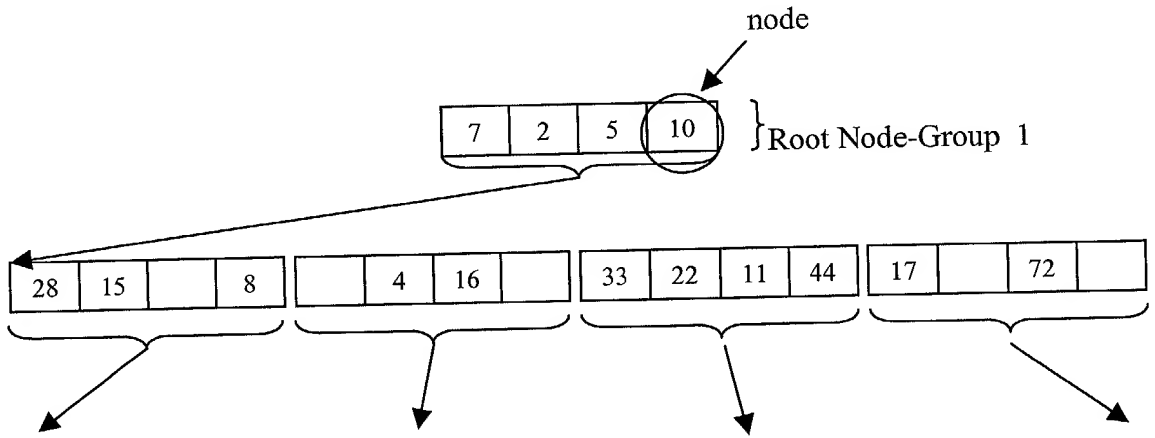
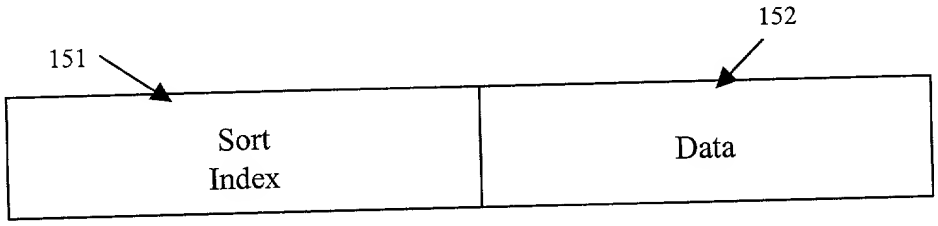


FIG. 14



150

FIG. 15

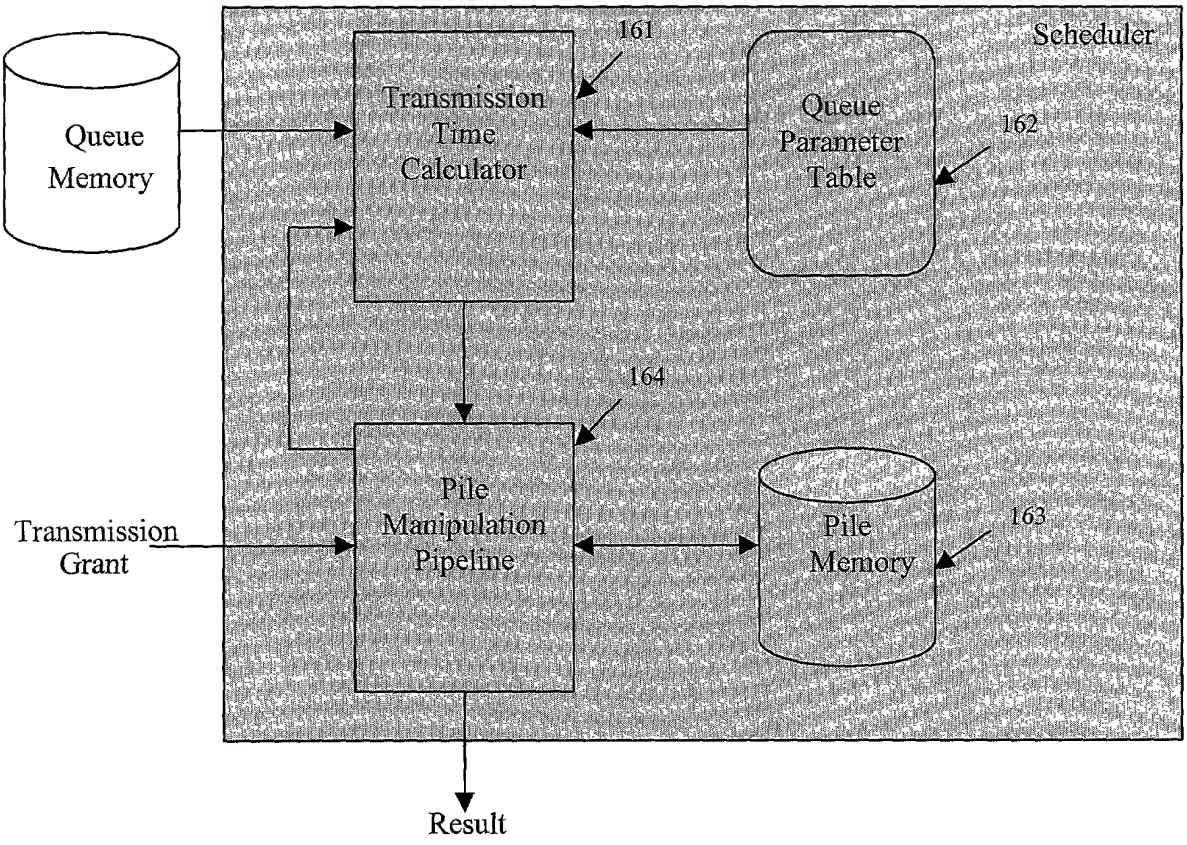
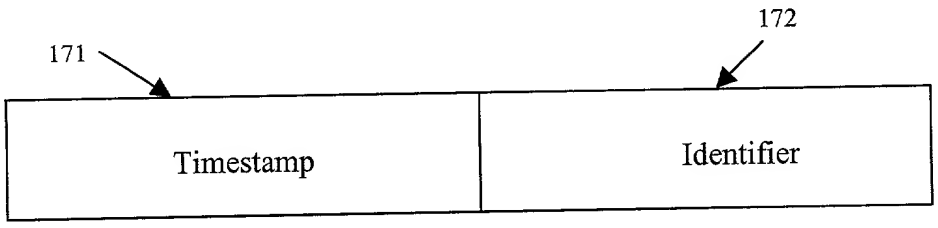
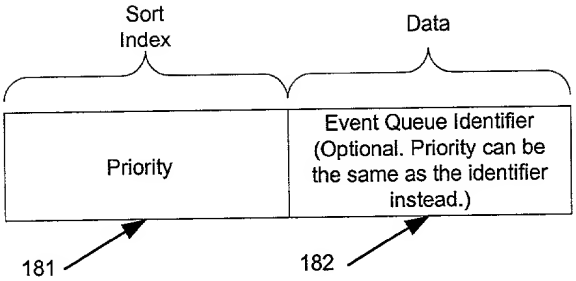

Scheduler
161
Transmission Time Calculator
Queue Memory
Queue Parameter Table
162
164
Pile Manipulation Pipeline
Pile Memory
163
Transmission Grant
Result
160

FIG. 16



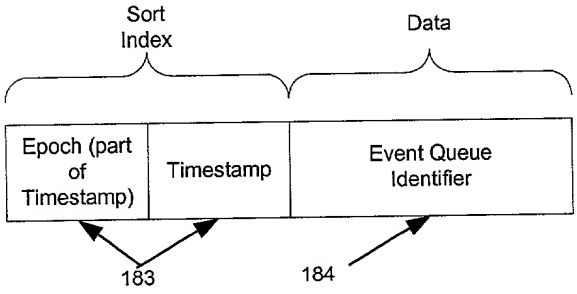
170

FIG. 17



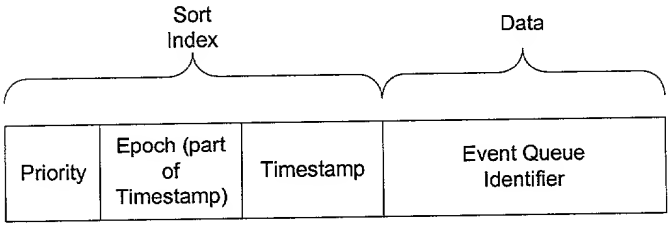
Pile Node Entry for Strict Priority

FIG. 18A



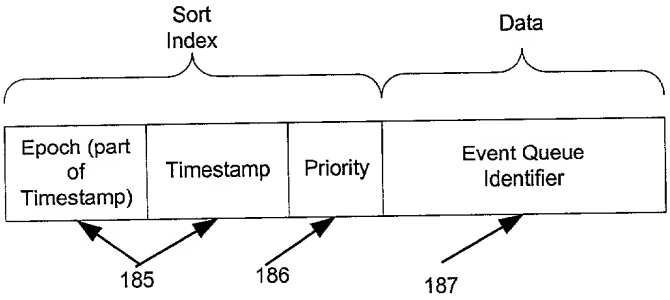
Pile Node Entry for Weighted Fair Queuing

FIG. 18B



Pile Node Entry for Queuing with Weighted Fair Priorities

FIG. 18C



Pile Node Entry for Traffic Shapping

FIG. 18D

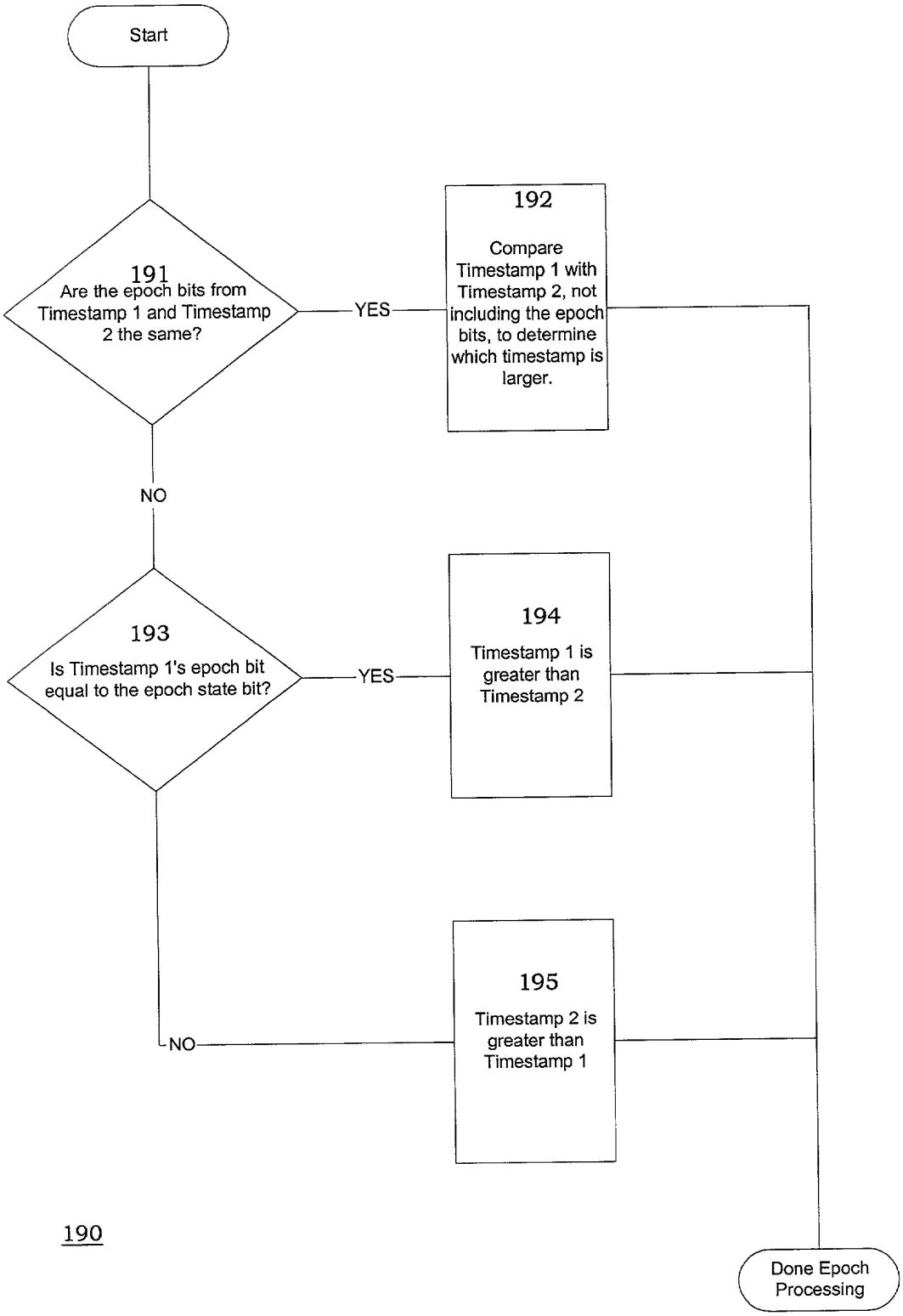

Start
191
Are the epoch bits from Timestamp 1 and Timestamp 2 the same?
YES
192
Compare Timestamp 1 with Timestamp 2, not including the epoch bits, to determine which timestamp is larger.
NO
193
Is Timestamp 1's epoch bit equal to the epoch state bit?
YES
194
Timestamp 1 is greater than Timestamp 2
NO
195
Timestamp 2 is greater than Timestamp 1
190
Done Epoch Processing

FIG. 19

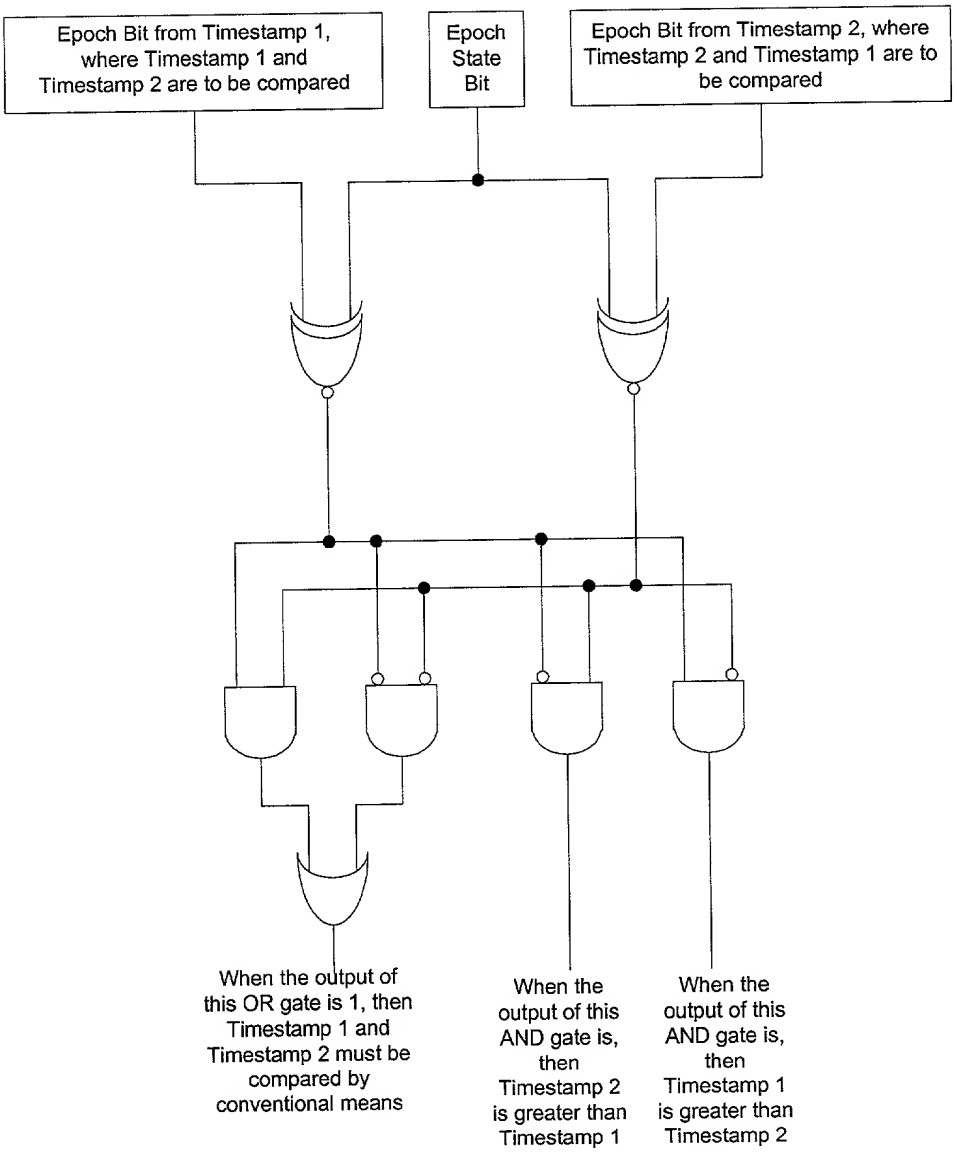

Epoch Bit from Timestamp 1, where Timestamp 1 and Timestamp 2 are to be compared
Epoch State Bit
Epoch Bit from Timestamp 2, where Timestamp 2 and Timestamp 1 are to be compared
When the output of this OR gate is 1, then Timestamp 1 and Timestamp 2 must be compared by conventional means
When the output of this AND gate is, then Timestamp 2 is greater than Timestamp 1
When the output of this AND gate is, then Timestamp 1 is greater than Timestamp 2
20

FIG. 20

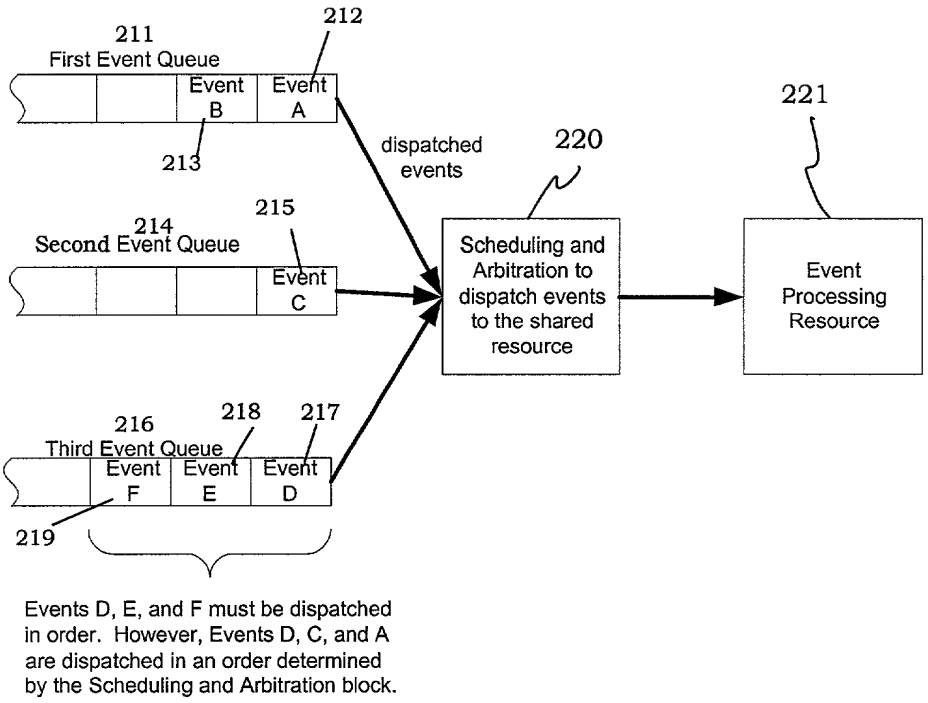

211
First Event Queue
212
Event B
Event A
213
dispatched events
220
221
214
Second Event Queue
215
Event C
Scheduling and Arbitration to dispatch events to the shared resource
Event Processing Resource
216
Third Event Queue
218
217
Event F
Event E
Event D
219
Events D, E, and F must be dispatched in order. However, Events D, C, and A are dispatched in an order determined by the Scheduling and Arbitration block.

FIGURE. 21